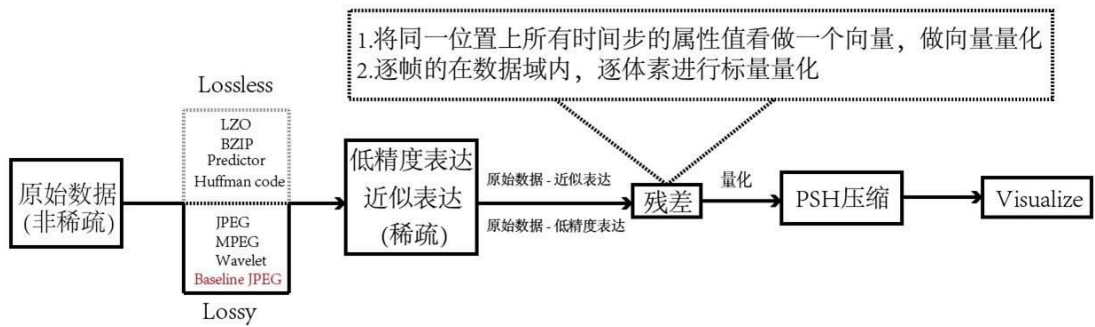


Weekly report

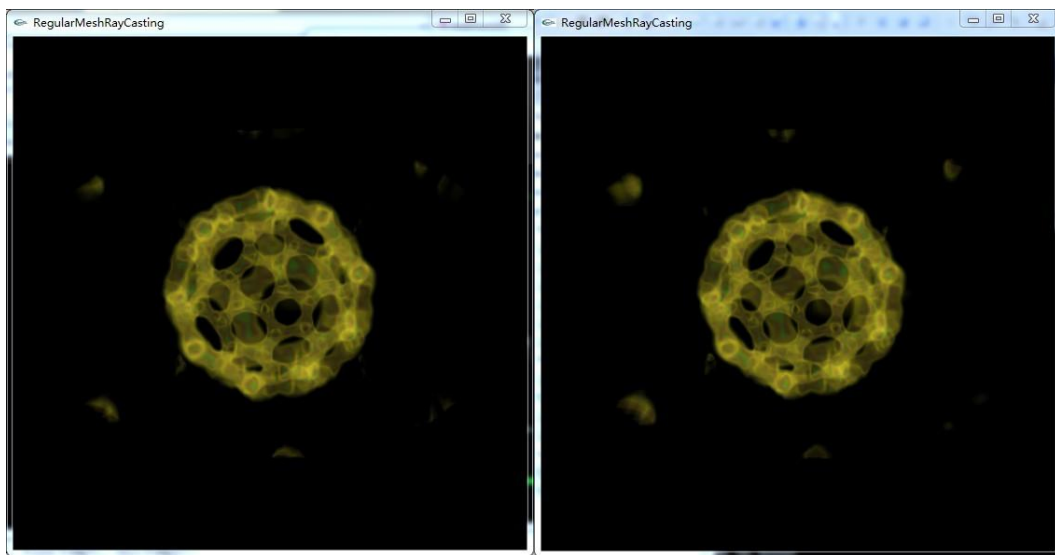
上周主要工作：

1. 完成 PSH 的修改。

依据之前的流水线，如下图



目前，完成了原有 PSH 压缩的修改，以及其与 Visualize 部分的融合。采用 Bucky（非稀疏）数据作为测试数据的绘制结果如下图



左图为使用 psh 方法的结果，右图为原始结果。

2. 针对流水线，调研了一些文献和资料，试图了解和掌握非稀疏体数据转为稀疏表达的方法。主要是 nvidia 的 VTC (Volume Texture Compression) 技术和 CEIG 2008 的论文<S3Dc: A 3Dc-based Volume Compression Algorithm>。后者的算法可以实现 8:1 的压缩比，并且

实现起来很直观简单，主要缺陷是该算法只能处理 RGBA 值，无法支持对 float 类型体数据的处理。之前想用 JPEG 等静态图像压缩格式和 MPEG 动态图像压缩格式进行体数据压缩，调研后发现实用的体纹理压缩算法必须要能高速、实时解压缩，不影响纹理拾取的速度，最好是不需解压就能直接用于渲染，所以 JPEG、Wavelets 等高压压缩率、低速度的方法不合适。其次，纹理压缩还需要支持纹理的随机访问，由于几乎无法预判像素的访问顺序，纹理压缩算法需要提供对解压缩纹理的快速随机访问。纹理压缩更注重压缩率，因此比图像压缩更能接受有损压缩算法。最后，纹理压缩对压缩和解压速度的要求是不对称的，压缩通常在预处理中完成，因此压缩过程的速度不是问题。

传统的 2D 纹理压缩，如 VQ (Vector Quantization)、S3TC、FXT1 等都是建立图形单位（多个图素单元组成）的索引表，再用索引表中的地址（索引号）来代替原数据，将数据量较大幅度减少，最大压缩率都在 $1/6 \sim 1/8$ 。区别在于，S3TC 和 FXT1 在纹理的多个部分建立不同的索引表，因此对小纹理的压缩效果同大纹理一样好；而 VQ 只为整个纹理建立一个索引表，纹理越大压缩率越高。

nVidia 的 3D 纹理压缩方法 VTC，是 S3TC 方法在三维的扩展，同样是一种硬件支持的压缩和解压缩方法，速度极快。依据它的 specification，最大压缩率也在 8:1，和 2D 的 S3TC 方法一致，此外，该技术也只支持 RGB (24bit) 和 RGBA (32bit)。

本周完成：

继续阅读相关文献进行调研，寻找和设计最佳的压缩方法和方案。力争完成整体方案的设计和实现构想。